The 2017 Visualization Technical Achievement Award

Jeffrey Heer

The 2017 Visualization Technical Achievement Award goes to Jeffrey Heer in recognition of his work on the design, development, dissemination, and popularization of languages for visualization. The IEEE Visualization & Graphics Technical Committee (VGTC) is pleased to award Jeffrey Heer the 2017 Visualization Technical Achievement Award.

Jeffrey Heer

University of Washington
Award Recipient 2017

Biography

Jeffrey Heer is an Associate Professor in the Paul G. Allen School of Computer Science & Engineering at the University of Washington. He received his B.S., M.S., and Ph.D. degrees in Computer Science from UC Berkeley. He then "betrayed" Berkeley to join the Stanford Computer Science faculty (2009–2013) before moving to UW (2013–). Heer is also a co-founder (with Sean Kandel and Joe Hellerstein) of Trifacta, a provider of interactive tools for scalable data transformation. At UW, Heer directs the Interactive Data Lab and conducts research on data visualization, human-computer interaction, and social computing. In addition to languages and software architectures for visualization, Heer's research interests include graphical perception, data wrangling, interactive machine learning, computational interpretation of chart images, natural language translation, and methods for collaborative data analysis.

Heer's interest in visualization started in his senior year of undergraduate studies at Berkeley, when he was exposed to the Hyperbolic Tree, a visualization technique developed at Xerox PARC. Enthralled by the mathematical elegance of the technique and the experience of "whipping through" massive hierarchies, he later sought out research on hierarchy visualization and worked on Degree-of-Interest Trees with Dr. Stuart Card and others at PARC. Heer's frustrations attempting to rapidly prototype different design ideas led him to explore higher-level abstractions for expressing visualization and interaction techniques.

This interest persisted when Heer returned to Berkeley for graduate studies, leading to the development of Prefuse, a Java toolkit that was among the first comprehensive software frameworks for information visualization. While developing Prefuse, Heer was warned on multiple occasions that the work he was doing was not "research" and that he might focus on more "scientific" pursuits. Stu Card helped Heer to look beyond the raw nuts and bolts of system building. Toolkits not only codify knowledge, but through their design instantiate a theory of doing, for example as reflected in a tool's "path of least resistance". Systems researchers should seek to articulate, motivate, and evaluate such theories. Through both user studies of programming tasks and feedback from real-world adoption, Heer continued his work on Prefuse over multiple years. Prefuse was first released as open source software in 2004 and published at ACM CHI 2005. The user community grew to thousands of developers across both industry and academia.

Meanwhile, the World Wide Web continued its ascent and it became clear that the web browser was the best vehicle for visualization to reach a wider audience. While Prefuse could run in the browser as a Java applet, applets were cumbersome and often required installation of custom plug-ins. In 2007 Heer developed Flare, a variant of Prefuse re-written for Adobe Flash, which at the time was commonly installed across browsers. This effort also provided Heer with an opportunity to explore richer support for animation, informed by his recent work on animated transitions with Dr. George Robertson at Microsoft Research. Flare was adopted by a number of designers and media firms, with visualizations reaching an audience of millions.

The next inflection point came when Heer joined the Stanford faculty and began working with Ph.D. student Mike Bostock. Bostock had the insight that, rather than focus primarily on data processing workflows (as Prefuse and Flare had done), why not instead start with a simple language of graphical marks, each consisting of a set of visual attributes that can be bound to data properties? In addition, native web technologies for rendering (e.g., scalable vector graphics) and interaction (via JavaScript) were maturing and becoming sufficiently performant. This confluence of ideas and good timing led to Protovis, a JavaScript framework for visualization featuring declarative constructs for visual encoding (*TVCG* 2009). Protovis soon took over as a leading visualization tool for the web, as Flash fell out of favor and JavaScript gained ascendancy. Refinement of Protovis continued for over a year, including the design of new animation abstractions that segment data into enter, exit, and update sets to control the behavior of dynamic elements as they come in and out of a visualization scene.

Based on experiences building, using, and supporting Protovis, Bostock (now in industry) developed and released a new project: D3.js. Instead of binding data elements to an intermediate abstraction (e.g., an internal scenegraph of graphical marks), why not remove the "middle man" and bind data objects directly to elements of the browser's

For more information, please visit https://homes.cs.washington.edu/~jheer/.

Document Object Model (DOM)? The expressiveness of the system would then be tied to the browser itself, rather than a fixed set of graphical mark types. From this idea came the abstraction of a selection that performs a data join between data and visual elements, expanding the notion of enter, exit, and update sets into a general pattern for data-driven documents (hence "D3"). The D3.js library (*TVCG* 2011) succeeded Protovis and – thanks to Bostock's engineering effort, community development, hundreds of examples, and later prominence as a graphics editor at The New York Times – has become a de facto standard for web-based interactive visualization.

After moving to Seattle, Heer and a new generation of students (including Arvind Satyanarayan, Kanit "Ham" Wongsuphasawat, Dominik Moritz, and Jane Hoffswell) have focused on building fully declarative languages for visualization, implemented on top of D3. The goal is to advance an ecosystem of interoperable tools for data analysis and visualization design underpinned by Vega, a declarative grammar that enables the creation of programs that generate interactive visualizations as output (ACM UIST 2014, *TVCG* 2016). Heer and collaborators have built upon the Vega foundation with higher-level tools such as Lyra, a graphical interface for designing custom visualizations for online publication without the need to write code (EuroVis 2014); Voyager, an exploratory analysis interface that integrates visual chart specification with a view recommender system (*TVCG* 2016, ACM CHI 2017), and Vega-Lite, a high-level grammar of interactive graphics that enables the rapid and concise specification of interactive multi-view visualizations (*TVCG* 2017). Vega has been adopted by commercial analytics vendors and popular data science environments, and can be used to add interactive visualizations to Wikipedia articles.

Award Information

The IEEE VGTC Visualization Technical Achievement Award was established in 2004. It is given every year to recognize an individual for a seminal technical achievement in visualization. VGTC members may nominate individuals for the Visualization Technical Achievement Award by contacting the awards chair, Holly Rushmeier, at vgtc-vis-awards@vgtc.org.